棋盘局面数据标定方法研究 *

丁　濛 a,b，张亦鹏 a,b，李淑琴 a,b

(北京信息科技大学 a.计算机学院; b.感知与计算智能联合实验室，北京 100101)

摘　要： AlphaGo 的成功使得深度学习方法在计算机博弈领域得到广泛关注。而基于深度学习模型的有监督训练依赖于大量高质量标定数据，但众多小众计算机博弈比赛棋种，存在缺少人类对局记录作为训练样本的问题,因此在使用深度学习模型前如何生成一个合理标定的局面数据集是值得研究探讨的问题。针对点格棋博弈问题，提出了一种数据哈希去重以及局面标定方法。根据不同阶段回合局面数据的特点，通过 Alpha-Beta 完全搜索、回溯标定、并行化 MCTS 算法标定以及对称扩展技巧，收集并标定不同回合数的点格棋局面样本。实验共获得了包含 15000000 个带标定点格棋局面样本的数据集，为基于深度学习模型的点格棋有监督训练提供了保障。此外，所提方法也为其他棋种训练数据的获取提供有价值的借鉴。

关键词： 数据标定；点格棋；棋盘局面；计算机博弈

中图分类号： TP181　　doi: 10.19734/j.issn.1001-3695.2018.08.0544

Study on chessboard configuration data calibration

Ding Meng[a,b], Zhang Yipeng[a,b], Li Shuqin[a,b]

(a. *School of Computer*, b. *Joint Labiratiry of Sensing & Computational Intelligence*, *Beijing Information Science & Technology University*, *Beijing* 100101, *China*)

Abstract: The success of AlphaGo has made deep learning methods widely concerned in the field of computer games. As we know, supervised training based on deep learning relies on a high-quality dataset consisting of a large amount of manually calibrated samples. However, many non-popular computer games are facing the problem of lacking human-game records as training samples. Therefore, how to generate a reasonably calibrated dataset of configuration data before using deep learning has significant value. In this paper, a data hashing and de-emphasis, and a configuration calibrated method are proposed for the dots and boxes game. According to the characteristics of configuration data at different stages, the proposed method makes use of full Alpha-Beta search, back-tracing search, parallel MCTS algorithm as well as symmetric flip extension to collect massive configuration data as training dataset. Experiment generates 15 million samples in total as the dataset to drive the supervised training model based on deep learning. In addition, the method proposed in this paper also provides valuable reference for the acquisition of training data of other chess games.

Key words: data calibration; dots and boxes; chessboard configuration; computer game

0　引言

计算机博弈就是让计算机学习人的思维模式,像人类一样,能够思维、判断和推理,作出理性决策,与人类选手或另一台计算机进行各种棋类的对弈[1]。它是人工智能领域的挑战性课题，是人工智能领域的重要研究方向。目前中国和国际上有很多的专家学者在开展计算机博弈研究,国际机器博弈协会(ICGA)每年组织一次计算机博弈大赛和学术研讨会,中国人工智能学会计算机博弈委员会每年也举行一次全国大学生计算机博弈大赛暨全国计算机博弈锦标赛，共设置了点格棋、苏拉卡尔塔棋、围棋、军棋、国际跳棋、二打一扑克牌(斗地主)和桥牌等 17 项棋牌类比赛项目[2]，极大推动了计算机博弈在世界范围内的发展。

2015 年 DeepMind 团队将深度学习技术引入计算机博弈之后[3~5]，集成深度学习[6]方法在计算机博弈领域得到了广泛关注与长足的发展。AlphaGo 使用的卷积神经网络本质上属于监督学习方法，一般来说好的学习模型的生成，需要大量的标定样本进行训练，才能使其具有良好的泛化效果。为此，如何为小众比赛棋种例如点格棋，在训练 CNN 前生成一个合理标定的局面数据集，是值得研究探讨的问题。本文主要以点格棋为抓手，深入研究棋盘局面数据的标定问题。

1　点格棋局面表示及哈希去重法

格棋是一种广为人知的双人棋类游戏，已经被纳入国际计算机奥林匹亚大赛和中国计算机博弈大赛多年。任意棋盘尺寸的点格棋规则如下：在一定大小的、均匀分布的矩形点阵中，两个玩家轮流在自己的回合中通过画水平或竖直方向的直线，连接相邻两点。若玩家画线后，任意一个由四个点围成的单位方格被封闭，即该格四周都被画上直线，画线玩家占领该格并得一分，随后得分玩家必须继续行动一回合。当无法再画线，即点阵中所有格子都被占领时，游戏结束，占领最多格子的玩家获胜。

本文采用 11×11 二维矩阵抽象地、格式化地表示了 5×5 点格棋各种局面中边、格子以及格子归属等信息。如图 1 左

收稿日期： 2018-08-12；**修回日期：** 2018-09-13　**基金项目：** 国家自然科学基金资助项目（61502039）；2017 年度教育教学改革研究专项招标课题（2017JGZB08）

作者简介： 丁濛（1982-），男，北京人，讲师，博士，主要研究方向为计算机博弈、图像处理（dmm@bistu.edu.cn）；张亦鹏（1993-），男，硕士研究生，主要研究方向为计算机博弈；李淑琴，女，教授，博士，主要研究方向为计算机博弈、人工智能.

上两个图所示。其中：标记 R 的块表示已被红方玩家占领的格子；标记 B 块表示已被蓝方玩家占领的格子；每个代表格子块上、下、左、右四个方向均有一个代表边的块，它们分别表示棋盘中与特定格的状态与该格相邻的四条边的状态。标记 1 表示已被占领的边；标记 0 表示未被占领的边。

在使用各种方法标记并收集点格棋样本数据时，都需要去除重复样本，提高数据集质量。为了节省空间开销，提高时间效率，记录局面样本时需要获得其高度压缩的、唯一的标志。本文采用建立样本哈希表的方式来记录已经收集的局面样本，高效地查表筛选收集到的局面样本，剔除其中的重复样本，记录未收集过的新样本。

具体方法是将点格棋局面采用 5 个属性唯一地描述：水平边占领状态、竖直边占领状态、红方玩家得分、蓝方玩家得分、当前回合归属。点格棋棋盘中有 6 行 5 列共 30 条水平边与 5 行 6 列共 30 条竖直边。逐行分别为水平边与竖直边编号，并使用 30 个比特位分别代表每个序号对应边的占领状态，则可以使用两个整型数分别描述特定点格棋局面的水平边占领状态与竖直边占领状态。特定点格棋局面到水平、竖直边占领状态的转换过程示例如图 1 所示。红方玩家得分、蓝方玩家得分、当前回合归属也可以分别由 3 个整型数表示。综上，任意一个点格棋局面可以由 5 个整型数组成的五元组唯一表示，该五元组则可以作为任意已收集局面样本的键，在哈希表中进行唯一标志。如图 1 所示局面对应的哈希表示形式为（123860205, 430258988, 1, 2, 0）。

双方玩家得分确定的情况下，双方玩家占领格子的具体位置分布可能不同，但此时不同格子占领状态的点格棋局面在局面评估与决策选取角度看是完全等价的。

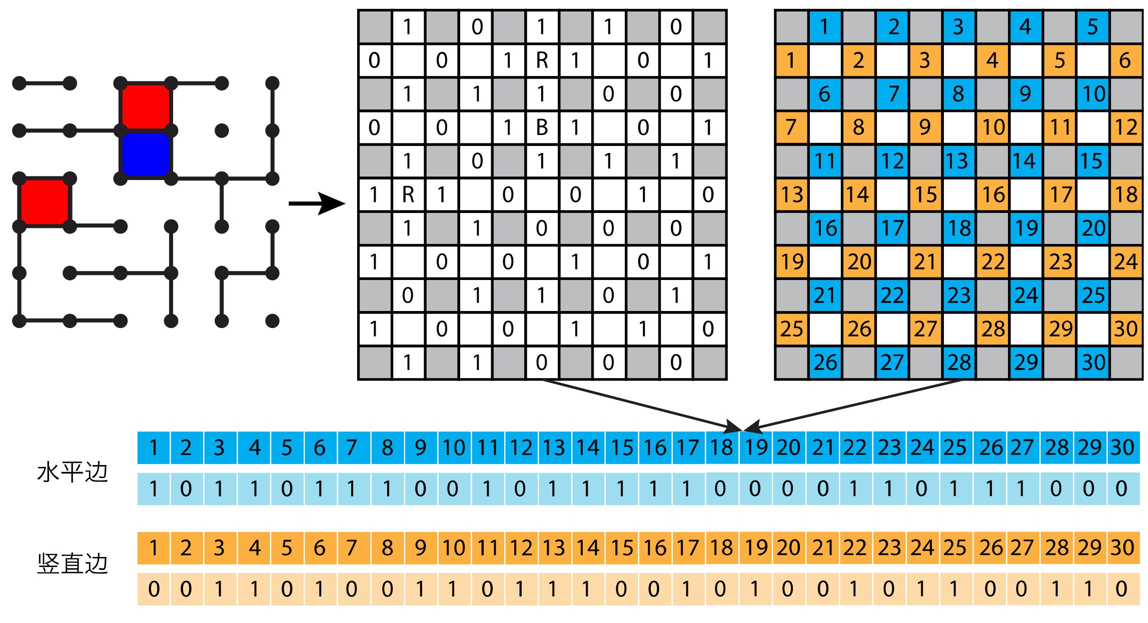


图 1　点格棋局面到水平、竖直边占领状态的转换过程示例

Fig. 1　Example of a conversion process from a checkerboard situation to a horizontal and vertical edge occupation state

2　数据标定方法的设计与实现

博弈过程中一般随着下棋的进程，局面类型及数量在不断变化，相应的也采用不同的策略。点格棋一般在 60 回合前结束。根据不同阶段回合局面数据的特点，本文提出第 28 至 59 回合的基于 Alpha-Beta 搜索的完全搜索标定法，第 23 至 30 回合的基于回溯标记算法的数据标定方法，第 0 至 24 回基于并行化 MCTS 算法的局面样本标定方法，以及基于对称翻转原理的数据扩充方法。

2.1　完全搜索数据标定法

基于 Alpha-Beta 剪枝[7]的极大极小值搜索被称为 Alpha-Beta 搜索，被广泛用于计算机博弈状态树的搜索中。由于 Alpha-Beta 是一种完全搜索策略，当搜索深度可以触及游戏结束的局面时，搜索结果将会是绝对正确的；另外，笔者通过大量实验发现，点格棋局面尤其是接近残局的局面下，决策近乎唯一。因此，可以将 Alpha-Beta 搜索结果作为当前局面下评分最高的决策，该决策可以作为样本数据的策略标定，而该决策的评分则可以作为当前局面的评估标定。

本文实现的 Alpha-Beta 搜索提供的局面评估值在[0,1]内，其中优势局面评估值为 1，劣势局面评估值为 0，评估值越大优势越大。令特定局面的评估值为 value，则局面回合归属进行翻转时，翻转的局面评估值为 1-value。

完全搜索数据标定法步骤如下：

a)随机产生局面作为当前局面；

b)如果当前局面对应回合数大于最小完全搜索回合数 nlimit 进入步骤 c)，否则回到步骤 a)；

c)对当前局面执行 Alpha-Beta 完全搜索，记录搜索深度 depth；

d)若 Alpha-Beta 完全搜索判定当前局面胜利进入步骤 e)，否则进入步骤 f)；

e)将当前局面价值标定为(1-depth)/(2*turn_sum)，将 Alpha-Beta 搜索确定的最佳着法选择概率标定为 1，其余着法选择概率标定为 0；

f)将当前局面价值标定为 depth/(2*turn_sum)，将 Alpha-Beta 搜索确定的最佳着法选择概率标定为 1，其余着法选择概率标定为 0。

其中，turn_sum 是游戏的最大回合数。

2.2　回溯数据标定法

考虑到由完全搜索进行局面评估并确定局面优劣势后，优势或劣势可以沿博弈树中树根到当前局面的必经路径向树根方向传递，并随着回合归属的翻转而翻转，递归地继续标定并收集局面样本。此外，当特定局面被评估为优势局面时，若其前驱局面的回合归属与该局面相同，则将也前驱局面也表示为优势局面。因为前驱局面一定有机会将局面引向优势

局面。

对于回合数比较靠前的局面，进行一次完全 Alpha-Beta 搜索的时间代价太大，而点格棋对局中可能出现某方玩家连续得分的情况，且连续得分期间回合归属不变，故在这种情况下，优势局面评估可以连续地向博弈树根节点方向传递。因此，本文提出优势和劣势两种情况下的回溯数据标记算法，具体算法的伪代码如图 2 所示：

Algorithm 1 Backtracing Alpha-Beta Search Algorithm

Input: *dataset*: a list of Alpha-Beta evaluated data;
Output: *new_dataset*: a list of newly evaluated data according to evaluations of data in *dataset*;

```
queue ← ∅
for each data in dataset do
    ENQUEUE(queue, data)
end for
while queue ≠ ∅ do
    board ← DEQUEUE(queue)
    precursors ← GET_PRECURSORS(board)
    for each precursor ∈ precursors do
        if precursors.size is 1 then
            if precursor.color is board.color then
                precursor.value ← board.value
            else
                precursor.value ← (1 − board.value)
            end if
        else if precursor.color is board.color then
            if board.value > 0.5 then
                precursor.value ← board.value
            end if
        else if board.value < 0.5 then
            precursor.value ← (1 − board.value)
        else
            continue
        end if
        APPEND(new_dataset, precursor)
        ENQUEUE(queue, precursor)
    end for
end while
```

图 2　回溯数据标记算法

Fig. 2　Backtracing alpha-beta search algorithm

其中，GET_PRECURSORS 方法按照规约的点格棋规则，获得特定局面的所有合法前驱。使用回溯标定算法时，局面评估通过传递规则获得，局面评估的传递路径，即是对应局面下的推荐决策，算法可以获得第 23 至 30 回合的大量带标定的点格棋局面样本。

2.3　并行 MCTS 数据标定法

第 23 回合以前点格棋局面十分难以通过完全搜索或回溯标定算法完成评估并标定，本文利用蒙托卡洛 MCTS 算法[8]来弥补这部分局面样本的空白。已被证明，随着模拟次数增加，MCTS 算法的局面评估与决策推荐结果收敛于极大极小值搜索[9]，然而该收敛过程十分缓慢，因为由 MCTS 算法提供准确的局面评估与决策推荐需要大量的模拟。

为了使用 MCTS 算法高效地标定足够多的局面样本，本文改进了朴素的 MCTS 算法。本文所述 MCTS 算法实现中使用哈希表数据结构存储博弈搜索树中的所有状态节点。该表以局面哈希值为键，以局面对应的博弈树节点的搜索状态为值。

本文在搜索状态索引表中添加表级线程互斥锁、节点级线程互斥锁，共两个粒度的线程互斥锁，并启动多个线程并行执行随机模拟过程，在线程间维护同一张搜索状态索引表。

表级线程互斥锁抢占机制如图 3 所示。

节点级线程互斥锁的读 / 写状态抢占机制步骤如下：

a) 当节点级线程互斥锁处于空闲状态，可以被写状态线程或读状态线程抢占；

b) 读 / 写状态线程都首先使用一轮循环等待确认线程互斥锁为空闲状态；

c) 读 / 写状态线程在确认互斥锁可用后立刻开始第二轮循环，尝试抢占互斥锁；

d) 若抢占失败，返回 b)；

e) 若读状态线程抢占互斥锁成功，将以互斥锁中的原子变量为读状态线程计数器，允许其他读状态线程共同占用互斥锁，不允许写状态线程占用互斥锁；

f) 在所有读状态线程释放对互斥锁的占用后（原子变量归 0 ），互斥锁回归空闲状态，允许被读 / 写状态线程抢占，即回到 a) 状态；

g) 若写状态线程抢占互斥锁成功，将以互斥锁中的原子变量为独占标志，不允许任何其他读 / 写状态线程占用互斥锁，直至该线程释放对互斥锁的占用；

h) 若节点级线程互斥锁在构造实例时配置成有退让的互斥锁，互斥锁占用状态下的状态节点更新请求将直接被无视；

i) 若节点级线程互斥锁在构造实例时配置成阻塞的互斥锁，互斥锁占用状态下的状态节点更新请求将被阻塞直至互斥锁回到空闲状态。

使用并行 MCTS 算法标定第 0 至 24 回合的点格棋局面样本时，MCTS 博弈树的根节点模拟胜率即为局面评估，根节点所有分支节点的模拟胜率则可作为决策评分。

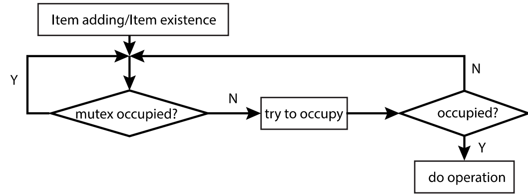


图 3　表级线程互斥锁抢占机制

Fig. 3　Table-level thread mutex lock preemption mechanism

2.4　对称翻转扩充数据法

点格棋棋盘为中心对称的正方形，故点格棋局面经过各种对称翻转仍与原局面等价。通过对已标定局面的对称翻转，可以成倍增加已标定的局面样本。由如图 4 可见，每种点格棋局面可以有 8 种对称翻转形式，故借助对称翻转，可以将已标定的点格棋局面样本扩充为原来数量的 8 倍，经过局面样本哈希表的过滤，仍能保留大量的增量局面样本。

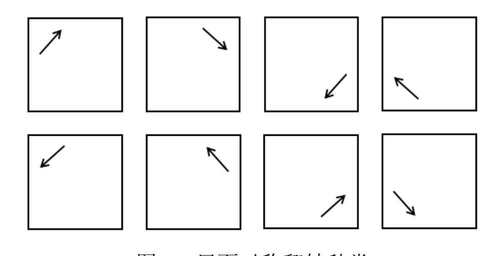


图 4　局面对称翻转种类

Fig. 4　Symmetric flip extension type.

3　实验

通过 Alpha-Beta 完全搜索、回溯标定、并行化 MCTS 算法标定以及对称翻转扩充，获得了共包含 15000000 个带标定点格棋局面样本的数据集。用 C++ 语言编译实现了 Alpha-Beta 搜索和基于 CNN 深度搜索算法，深度模型使用 Caffe 框架实现。实验环境为：用 g++ 5.4.0 编译器，Ubuntu 16.04 x64 系统，Intel® Core™ i7-6700HQ CPU @ 2.60 GHz。

本实验主要通过实例的方式，展示本文所述样本的标注与扩展方法。

3.1　搜索标注实例

选取图 5 所示局面样本，该样本对应局面轮到蓝方行动，该局面的哈希表示形式为（758027384, 113572561, 3, 2, 1）。该样本的 Alpha-Beta 完全搜索标注结果为：胜率 77.5%，最佳走法（type: vertical, row: 1, col: 4）。

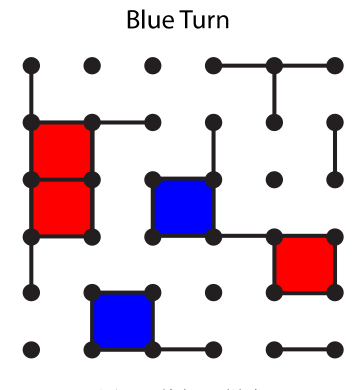


图 5　待标注样本

Fig. 5　Sample to be calibrated

3.2　回溯标注实例

根据原样本的标注结果，可以回溯标注数百个新样本，其中一种针对图 5 局面样本回溯的样本如图 6 所示。图 6 局面的哈希表示形式为（758027384, 113556177, 3, 1, 1）。本文将图 6 所示样本与原样本对应局面标示有相同的蓝方胜率，其最佳走法为（type: vertical, row: 2, col: 2），该样本的 MCTS 标注结果为：胜率 71.39%。

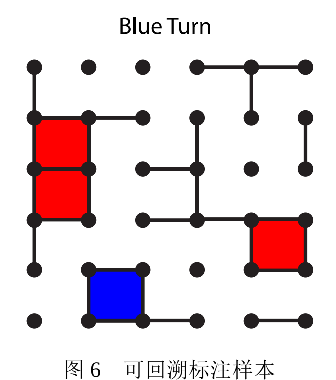
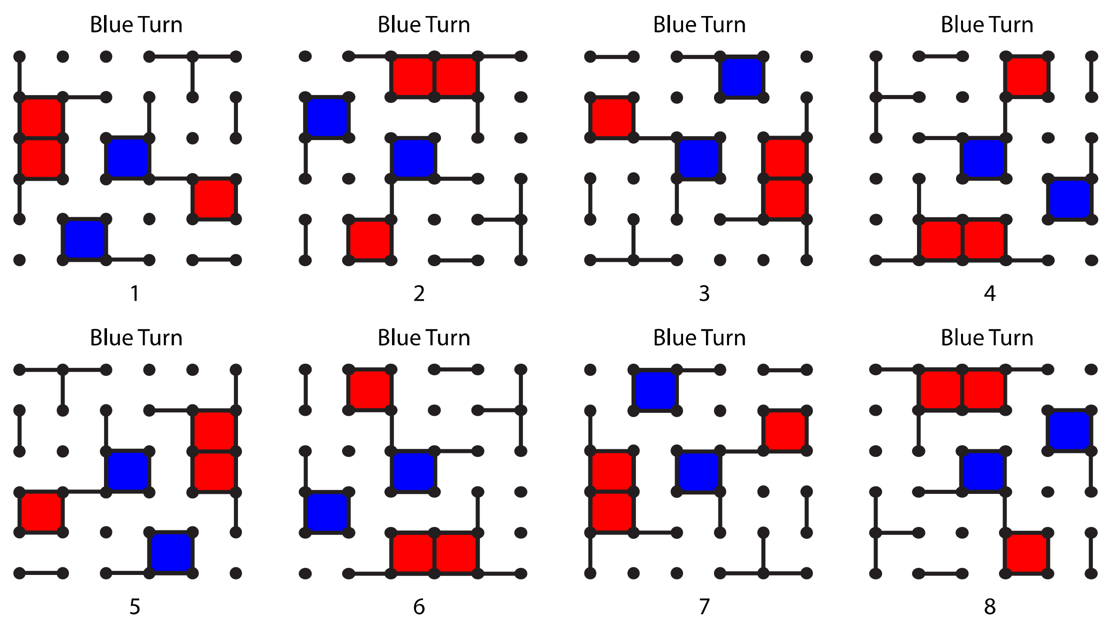


图 6　可回溯标注样本

Fig. 6　Sample calibrated by back-tracing

3.3　对称翻转扩充实例

经过对称翻转扩充，可以得到 7 个与原样本完全等价的局面样本。如图 7 所示，其中 0 号局面为原样本图 5 所示局面。本文将图 7 所示样本与原样本对应局面标示有相同的胜率，即蓝方胜率 77.5%。



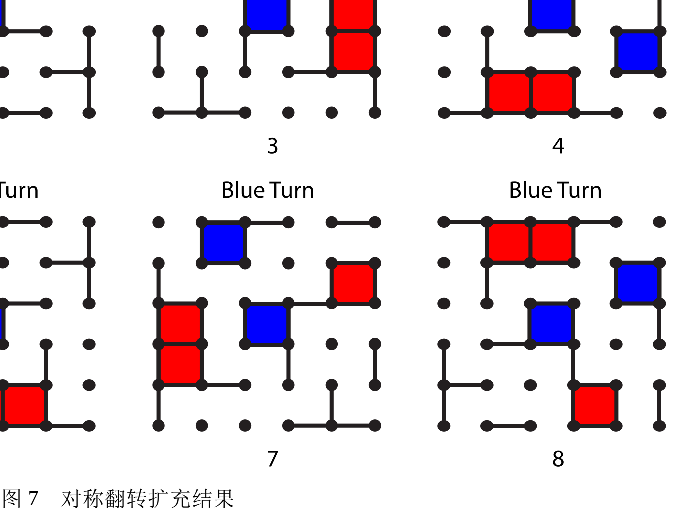


图 7　对称翻转扩充结果

Fig. 7　Results of Symmetric flip extension.

4　结束语

本文通过分析点格棋的规则特点，提出了一系列样本数据标定、扩充、去重方法，包括：基于压缩规则的 Alpha-Beta 搜索的完全搜索标定法，基于回溯标记算法的数据扩充方法，基于并行化 MCTS 算法的局面样本标定方法，基于对称翻转原理的数据扩充方法。有效获取到不同回合游戏的样本数据并有效标定，一方面为深度学习模型的有监督训练提供了保障，另一方面希望为其他小众比赛棋种训练数据的获取提供帮助。

参考文献：

[1] Wang Yajie, Qiu Hong kun, el al. Computer game competition and reform of innovative talent training mode [J]. Experimental Technology and Management, 2016, 33(10): 10-14

[2] 中国大学生计算机博弈大赛组委会. 全国计算机博弈竞赛总则 [EB/OL]. [2018-08-12]. http://computergames. caai. cn/jsgz00. html. (China University Students Computer Game Competition Organizing Committee. General rules of national computer game competition[EB/OL].[2018-08-12]. http://computergames. caai. cn/jsgz00. Html.)

[3] Mnih V, Kavukcuoglu K, Silver D, *et al.* Human-level control through deep reinforcement learning [J]. Nature, 2015, 518(7540): 529-533.

[4] Silver D, Huang A, Maddison C J, *et al.* Mastering the game of Go with deep neural networks and tree search [J]. Nature, 2016, 529(7587): 484-489.

[5] Silver D, Schrittwieser J, Simonyan K, *et al.* Mastering the game of Go without human knowledge. [J]. Nature, 2017, 550(7676): 354.

[6] Qiu Xueheng, Zhang Le, Ren Ye *et al*. Ensemble deep learning for regression and time series forecasting [C]//Proc of IEEE. Symposium on Computational Intelligence in Ensemble Learning.2014: 1-6.

[7] Pearl J. The solution for the branching factor of the alpha-beta pruning algorithm and its optimality [J]. Communications of the ACM, 1982, 25 (8): 559-564.

[8] Chaslot G, Bakkes S, Szita I, *et al*. Monte-Carlo Tree Search: A New Framework for Game AI [C]// Proc of the 4th Artificial Intelligence and Interactive Digital Entertainment Conference. AAAI Press, 2008: 216-217.

[9] Bouzy B. Old-fashioned computer go vs monte-carlo go [C]//Proc of IEEE. Symposium on Computational Intelligence and Games.2008.